UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/784,374	02/23/2004	Pal Takacsi-Nagy	BEAS-01389US2	8929

23910 7590 07/27/2007
FLIESLER MEYER LLP
650 CALIFORNIA STREET
14TH FLOOR
SAN FRANCISCO, CA 94108

EXAMINER	
WANG, JUE S	
ART UNIT	**PAPER NUMBER**
2193	
MAIL DATE	**DELIVERY MODE**
07/27/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/784,374	TAKACSI-NAGY ET AL.	
	Examiner	Art Unit	
	Jue S. Wang	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *23 February 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-39* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-39* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *23 February 2004* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *See Continuation Sheet*.	6) ☐ Other: _____.

Continuation of Attachment(s) 3). Information Disclosure Statement(s) (PTO/SB/08), Paper No(s)/Mail Date :05/19/2004, 01/24/2005, 03/07/2005, 04/01/2005, 09/21/2006, 05/2 3/2007.

DETAILED ACTION

1. Claims 1-39 have been examined.

Specification

2. The specification is objected to because of the following informalities:

3. The use of the trademark Java has been noted in this application. It should be capitalized wherever it appears and be accompanied by the generic terminology.

Although the use of trademarks is permissible in patent applications, the proprietary nature of the marks should be respected and every effort made to prevent their use in any manner which might adversely affect their validity as trademarks.

Drawings

4. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(4) because reference characters "118" and "108" have both been used to designate Handle MIS Reply in Figure 1. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. Each drawing sheet submitted after the filing date of an application must be labeled in the top margin as either "Replacement Sheet" or "New Sheet" pursuant to 37 CFR 1.121(d). If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Claim Rejections - 35 USC § 112

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claims 2, 9, 10, 13, 14, 19, 26, 27, 30, 31 36, and 37 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

A. The following lacks antecedent basis in the claims:

i. Claim 13, "said software component" in line 2. It is not clear whether "said software component" refers to the "first software component" or "second software component" recited in claim 12.

ii. Claim 30, "said software component" in line 2. It is not clear whether "said software component" refers to the "first software component" or "second software component" recited in claim 29.

B. The following claim language is not clear and indefinite:

i. As per claim 9, lines 3-4, and claim 26, lines 3-4, the phrase "compensating software activity" is use. This limitation is not clearly understood because it is not clear what functionality is performed by the compensating software activity (i.e., does the compensating software activity handle exceptions that are thrown during the transaction, or does the compensating software activity correct non-critical errors that might occur during the transactions, or does the compensating software activity undo all previous actions associated with a transaction in the event of an error?)

ii. As per claim 14, lines 2-3, and claim 31, lines 2-3, the phrase "said action construct further allows said second software component callback the said first software component" is used. This limitation is not clearly understood because it is not clear how the second software component (written in the existing language) can communicate with the first software component (written in a language that includes the existing language further extended with additional constructs not found in the existing language) since the execution for the existing language would not normally understand the extended existing language with the additional constructs.

C. Claims 2, 13, 19, 30, 36, 37 contain the trademark/trade name Java. Where a trademark or trade name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of 35 U.S.C. 112, second paragraph. See *Ex parte Simpson*, 218 USPQ 1020 (Bd. App. 1982). The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. A trademark or trade name is used to identify a source of goods, and not the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case, the trademark/trade name Java is used to identify/describe a programming language and, accordingly, the identification/description is indefinite.

Appropriate corrections are required.

Any claim not specifically addressed, above, is being rejected as incorporating the deficiencies of a claim upon which it depends.

Claim Rejections - 35 USC § 101

7. 35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

8. Claims 18-34, 37, and 39 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

9. Claim 18 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. In claim 18, a "system" is recited; however, it appears that the system would reasonably be interpreted by one of ordinary skill in the art as software, per se, since the existing programming language and the means for extending an existing programming language would reasonably be interpreted by one of ordinary skill in the art as software, per se. As such, it is believed that the system of claim 18 is reasonably interpreted as functional descriptive material, per se, failing to be tangibly embodied or include any recited hardware as part of the device.

10. Claims 19-34 fail to resolve the deficiencies of claim 18. Claims 19-34 disclose additional features of language constructs recited in claim 18. The limitations recited in claims

19-34 do not invalidate the reasonable interpretation of the system as software, per se, by one of ordinary skill in the art.

11. Claim 37 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. In claim 37, a "system" is recited; however, it appears that the system would reasonably be interpreted by one of ordinary skill in the art as software, per se, since the Java programming language and the means for extending Java programming language would reasonably be interpreted by one of ordinary skill in the art as software, per se. As such, it is believed that the system of claim 37 is reasonably interpreted as functional descriptive material, per se, failing to be tangibly embodied or include any recited hardware as part of the device.

12. Claim 39 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. In claim 39, the "computer program product" recited is a program which is software, per se.

Claim Rejections - 35 USC § 102

13. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

14. Claims 1-3, 18-20 and 35-39 are rejected under 35 U.S.C. 102(a) as being anticipated by Plaindoux, "XML transducers in Java".

15. As per claim 1, Plaindoux teaches the invention as claimed, including a method for extending an existing programming language, comprising the steps of:

selecting an existing programming language; and,

extending an existing programming language by adding at least one language construct defined by a second language ("high-level integration of XML to Java", see abstract, paragraph 2, and section 1, paragraphs 2, 4).

16. As per claim 2, Plaindoux further teaches that the existing programming language is Java (see abstract, paragraph 2, and section 1, paragraphs 2, 4).

17. As per claim 3, Plaindoux further teaches that the second language is XML (see abstract, paragraph 2, and section 1, paragraphs 2, 4).

18. As per claims 18-20, these are the system claims of claims 1-3. Therefore, they are rejected using the same reasons as claims 1-3.

19. As per claim 35, this is the system claim of claim 1. Therefore, it is rejected using the same reasons as claim 1.

20. As per claim 36, Plaindoux teaches the invention as claimed, including a method for extending Java programming language, comprising the steps of:

selecting Java programming language; and,

extending Java programming language by adding at least one language construct defined by XML ("high-level integration of XML to Java", see abstract, paragraph 2, and section 1, paragraphs 2, 4).

21. As per claim 37, this is the system claim of claim 36. Therefore, it is rejected using the same reasons as claim 36.

22. As per claim 38, Plaindoux teaches the invention as claimed including an existing programming language extended with at least one language construct defined by a second language ("high-level integration of XML to Java", see abstract, paragraph 2, and section 1, paragraphs 2, 4).

Plaindoux does not teach creating a program using the extended existing programming language. However, it is inherent that a program is created using the extended existing programming language because programming languages are developed solely for the purpose of creating programs.

23. As per claim 39, this is the computer program product claim of claim 38. Therefore, it is rejected using the same reasons as claim 38.

Claim Rejections - 35 USC § 103

24. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

25. Claims 4-6, 12-14, 21-23, and 29-31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Plaindoux, "XML transducers in Java", in view of Meredith, (US 6,516,322 B1).

26. As per claim 4, Plaindoux does not teach a parallelism construct representing parallel branch of program execution.

Meredith teaches a scheduling language SLANG which is written in XML, and includes syntax that allows for expression of features associated with the model for describing distributed and parallel computing (see column 2, lines 14-20, column 6, lines 26-38, and column 12, lines 45-64), where SLANG has a parallelism construct representing parallel branch of program execution (the task construct, see Figs 16b-16d, column 13, lines 3-5, 46-54, and column 14, lines 57-67).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux such that the XML integrated into Java contains construct to specify workflow including a parallel construct representing parallel branch of program execution as taught by Meredith because the XML constructs provide compositional specification of autonomously executing systems such that the cost of developing and managing applications that span business units connected by communications networks is greatly reduced (see column 6, lines 26-55 of Meredith).

27. As per claim 5, Plaindoux does not teach that the parallelism construct further comprises a plurality of branch constructs defined by said second language, wherein said branch constructs represent parallel branches of program execution comprising of at least one software activity.

Meredith teaches that the parallelism construct further comprises a plurality of branch constructs defined by said second language, wherein said branch constructs represent parallel branches of program execution comprising of at least one software activity (the action construct, see column 13, lines 3-5, 46-55, column 14, lines 16-24, 57-67).

28. As per claim 6, Plaindoux does not teach said parallelism construct is further nested within a similar parallelism construct.

Meredith teaches that the parallelism construct is further nested within a similar parallelism construct (the task construct is nested within the partition construct which is also a parallelism construct, see Figs 6, 16b-16d, 23b, column 14, lines 57-67, column 15, lines 54-61).

29. As per claim 12, Plaindoux does not teach that said language construct is an action construct representing an activity that allows a first software component written using the extended existing programming language to call an operation on a second software component written using said existing programming language.

Meredith teaches that the language SLANG has the functionality a first software component written using the programming language XML to call an operation on a second software component written using another programming language such as Java (see column 13, lines 3-5, 46-55; specifically "the action can be mapped to invocations on common object model (COM) objects, ... or other native technology behavior"). While Meredith does not specifically teach a construct to do the mapping, it would have been obvious to one of ordinary skill in the art at the time of the invention to extend SLANG with such a construct to denote the mapping since XML itself is an extensible language. Furthermore, while Meredith does not explicitly teach that the second software component is written in Java, it would have been obvious that the native technology behavior includes software components written in Java since Java is a popular programming language with the advantage of being platform independent.

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux such that the XML integrated into Java contains constructs to specify workflow including an action construct representing an activity that allows a first software component written using the extended programming language (XML) to call an operation on a second software component written using another programming language as taught by Meredith because the XML constructs provide compositional specification of autonomously executing systems such that the cost of developing and managing applications that span business units connected by communications networks is greatly reduced (see column 6, lines 26-55 of Meredith).

30. As per claim 13, Plaindoux and Meredith do not specifically teach that the action constructs allows said software component to call a piece of Java code. However, Meredith teaches that "the action can be mapped to invocations on common object model (COM) ... or other native technology behavior" (see column 13, lines 3-5, 46-55). It would have been obvious to one of ordinary skill in the art at the time of the invention that the native technology behavior includes software components written in Java since Java is a popular programming language with the advantage of being platform independent.

31. As per claim 14, Plaindoux and Meredith do not specifically teach that the action construct further allows said second software component call back the said first software component. While Meredith only discloses mapping the action to invocations on a software object (see column 13, lines 3-5, 46-55), it would have been obvious to one of ordinary skill in the art at the time of the invention that there must also be a mapping between invoked software components back into actions which consist of a message in order to communicate its status such as reporting its completion.

32. As per claims 21-23 and 29-31, theses are the system claims of claims 4-6 and 12-14. Therefore, they are rejected using the same reasons as claim 4-6 and 12-14.

33. Claims 7, 8, 11, 24, 25, and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Plaindoux, "XML transducers in Java", in view of Mital et al. (US 7,184,867 B1, hereinafter Mital).

34. As per claim 7, Plaindoux does not teach that said language construct is a transaction construct representing transaction block of at least one software activity.

Mital teaches a tool to allow a user to create a schedule for business workflow where the schedule can be converted to a scheduling programming language written in XML (SLANG), where the model allows the user to define transactions representing transaction block of at least one software activity which are translated to the context constructs (see Fig 2, Fig 3, Fig 24, column 2, lines 29-46, column 9, lines 47-60, and column 16, lines 40-65).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux such that the XML integrated into Java contains constructs to specify workflow including a transaction construct representing transaction block of at least one software activity as taught by Mital because the XML constructs are used to specify workflow which facilitates the processing of business transactions between different companies (see column 1, line37 – column 2, line 46 of Mital).

35. As per claim 8, Plaindoux does not teach that the transaction construct further specifies the number of retry attempts to perform the software activities inside said transaction block.

Mital that the transaction construct of SLANG further specifies the number of retry attempts to perform the software activities inside said transaction block (see column 2, lines 29-46, column 9, lines 47-60, column 13, lines 8-9 and Table 2, and column 16, lines 40-65; EN: while Mital does not explicitly state as such, it would have been obvious that the transaction construct specifies the number of retry attempts since retry count is one of the properties of the transaction shape and the model is converted to SLANG).

36. As per claim 11, Plaindoux does not teach said language construct is an exception handler construct representing an execution mechanism comprising of exception handler construct defined by said second language, which represents exception not caught by the programming language handler methods.

Mital teaches a tool to allow a user to create a schedule for business workflow where the schedule can be converted to a scheduling programming language written in XML (SLANG), where the model allows the user to define an exception handler construct representing an execution mechanism comprising of exception handler construct (see column 2, lines 29-46, column 9, lines 47-60, and column 16, lines 40-65; EN: while Mital does not explicitly state that SLANG has an exception handler, it would have been obvious that SLANG has an exception handler since the transaction shapes have catch pages associated to define routines invoked on a failed transaction, and the model is converted to SLANG).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux such that the XML integrated into Java contains constructs to specify workflow including an exception handler construct representing an execution mechanism comprising of exception handler construct (EN: the XML construct would not be caught by Java since it's defined in XML) as taught by Mital because the XML constructs are used to specify workflow which facilitates the processing of business transactions between different companies (see column 1, line37 – column 2, line 46 of Mital).

37. As per claims 24, 25, and 28, these are the system claims of claims 7, 8, and 11. Therefore, they are rejected using the same reasons as claims 7, 8, and 11.

38. Claims 9, 10, 26, and 27 are rejected under 35 U.S.C. 103(a) as being unpatentable over Plaindoux, "XML transducers in Java", in view of Mital et al. (US 7,184,867 B1, hereinafter Mital), as applied to claims 7 and 24 above, further in view of Alonso et al. "Advanced Transaction Models in Workflow Contexts", (hereinafter Alonso).

39. As per claim 9, Plaindoux and Mital do not teach that said transaction construct is further enclosed within a saga construct comprising of compensation construct with at least one compensating software activity, wherein the saga construct represents a long running transaction.

Alonso teaches a method to convert high level specifications of advanced transaction models such as linear sagas for long lived transactions with compensation activity into workflow processes (see pages 4-5, section 4, paragraphs 1, 2, and section 4.1).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux and Mital to implement linear sagas using the preexisting XML constructs used to define workflow including compensations for transactions (see column 9, lines 59-63, column 50, lines 50-56) as taught by Alonso because linear sagas provide a model to deal with long lived transactions and allow a transaction to release resources before committing (see page 4, right column, last paragraph of Alonso). Furthermore, while Plaindoux, Mital, and Alonso do not teach a saga construct to denote the saga, it would have been obvious to one of ordinary skill in the art at the time of the invention to extend SLANG with such a construct to mark the saga functionality implemented since XML itself is an extensible language.

40. As per claim 10, Plaindoux and Mital do not teach that said saga construct further comprises a plurality of transaction blocks. However, Alonso teaches that the linear saga comprises a plurality of transaction blocks (see page 6, Figure 1), so the saga construct of Plaindoux modified by Mital and Alonso would have a plurality of transaction blocks.

41. As per claims 26 and 27, these are the system claims of claims 9 and 10. Therefore, they are rejected using the same reasons as claims 9 and 10.

42. Claims 15-17 and 32-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Plaindoux, "XML transducers in Java", in view of van der Aalst, "XML Based Schema Definition for Inter-Organization Workflow", (hereinafter Aalst).

43. As per claim 15, Plaindoux does not teach that said language construct is a multiple receive construct that allows a software component written using the extended existing programming language to wait on multiple input events received.

Aalst teaches the language XRL which is based on XML syntax and provides support for routing of workflow among trading partners for internet based electronic commerce services (see page 1, section 1, paragraph 1), where XRL has a multiple receive construct that allows a software component to wait on multiple input events received (the wait_any construct, see pages 16-17, section 4.7).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux such that the XML integrated into Java contains constructs to specify workflow including a multiple receive construct that allows a software component written using the extended existing programming language to wait on multiple input events received as taught by Aalst because the XML constructs provide support for routing of workflow among trading partners to Internet based electronic commerce services to facilitate increased productivity and interoperability (see page 1, section 1, paragraph 1 of Aalst).

44. As per claim 16, Plaindoux do not teach that said multiple receive construct further allows said software component proceed on a particular branch of program execution, based on the input event that occurred first within the said multiple input events.

While Aalst does not teach that the multiple receive construct allows software component proceed on a particular branch of program execution, based on the input event that occurred first within the said multiple input events. However, it would have been obvious that this functionality can be achieved in conjunction with the condition construct (see page 16, section 4.6, page 24, paragraph 2, and page 26, Figure 12) by placing the condition construct directly after the wait_any construct such that different branches are taken depending on the event that was received by wait_any. Furthermore, while Plaindoux, and Aalst do not teach the multiple receive construct has the specified functionality, it would have been obvious to one of ordinary skill in the art at the time of the invention to augment the multiple receive construct of XRL with such a functionality implemented since XML itself is an extensible language.

45. As per claim 17, Plaindoux does not teach that said construct is a looping construct with ordering of messages received, representing looping functionality, wherein the order allows said messages to be received in an order.

Aalst teaches the language XRL which is based on XML syntax and provides support for routing of workflow among trading partners for internet based electronic commerce services (see page 1, section 1, paragraph 1), where XRL has a looping construct representing looping functionality (see pages 16-17, section 4.7, the while_do construct). While Aalst does not specifically teach that the loop construct has an ordering of messages received, where the ordering allows said messages to be received in an order, it would have been obvious that an ordering of the messages received does not need to rely on a looping construct, and instead can be achieved from the content of the loop. For example, in the mail order processing example presented on pages 24 –27, section 6, the while_do loop is used to find a shipper where a list of shippers is contacted in order to find an available shipper. Since a message is received to determine whether the shipper can perform the shipment and the shippers are contacted in order, the messages are received in order because the shippers are contacted in order according to a list (see page 24, paragraph 2). Furthermore, while Aalst do not teach the loop construct has the specified functionality, it would have been obvious to one of ordinary skill in the art at the time of the invention to augment the loop of XRL with such a functionality since XML itself is an extensible language.

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Plaindoux such that the XML integrated into Java contains constructs to specify workflow including a looping construct with ordering of messages received, representing looping functionality, wherein the order allows said messages to be received in an order as taught by Aalst because the XML constructs provide support for routing of workflow among trading partners to Internet based electronic commerce services to facilitate increased productivity and interoperability (see page 1, section 1, paragraph 1 of Aalst).

46. As per claims 32-34, these are the system claims of claims 15-17. Therefore, they are rejected using the same reasons as claims 15-17.

Conclusion

47. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

- Lucas et al. (US 6,754,884 B1) is cited to teach programming language extensions for processing XML objects and related applications.
- Ankireddipally et al. (US 6,971,096 B1) is cited to teach transaction data structure for process communications among network-distributed applications.
- Bau III (US 7,043,722 B2) is cited to teach mixed language expression loading and execution methods and apparatuses.
- Vion-Dury et al. (US 7,240,331 B2) is cited to teach bi-valuation of programming statements.
- Bosworth et al. (US 2004/0040011 A1) is cited to teach multi-language execution method.

48. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jue S. Wang whose telephone number is (571) 270-1655. The examiner can normally be reached on M-Th 7:30 am - 5:00pm (EST).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

J.W.
7/20/2007

[illegible]
MENG-AI T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100